REMARKS

Claims 1-63 are pending in the present patent application. The Examiner has rejected claims 1-63. Applicant has amended claims 1, 2, 8, 14, 16, 22, 23, 35, 37, 42, 43, 44, 48, 50, 56, 58 and 63. Applicant respectfully requests reconsideration of claims 1-63 in view of at least the following amendments and remarks.

I. Objections to Drawings

The Examiner objected to the Drawings stating reference characters "320, 321 and 322 have been used to designate records in 301 and 302 lookup tables, and said references were not mentioned in the specifications.

Applicant has amended the drawing in Figure 3, and submits a hand-corrected copy of the Figure 3. Formal drawings will be submitted upon allowance of the present case.

II. Objections to Claims 14, 35, 42, 48, 56 and 63

The Examiner has objected to claims 14, 35, 42, 48, 56 and 63 because of various typographical errors.

Applicant has amended claims 14, 35, 48 and 56 to correct the dependency references. The Examiner pointed to claims 42 and 63 containing the word "effected" and proposed a replacement with the word "affected." Claims 42 and 63 contain the word "effects." Applicant assumes the Examiner meant to refer to the latter word. Applicant has amended claims 42 and 63 to replace the word "effects" with the phrase "should effect" to emphasize the use of the verb "effect" in the context of the invention. As described in the Applicant's argument below, the updates to data records in lookup tables may effect (in the

sense of produce) a new array of bits, the binary values of each is determined by the rank value associated with a record in a lookup table. Therefore, the use of "effecting an update" on an bit array seems more appropriate to describe the action of updating the bit array by replacing (as in a replacing a single entry of an index), or by flipping bit at specific positions to reflect the changes due to the updates in the lookup tables.

III. Rejections of Claims 1-15, 19, 22-28, 40, 43-57 and 61 Based on 35 U.S.C. § 112

The Examiner has rejected claims 1-15, 19, 22-28, 40, 43-57 and 61 under 35 USC 112, second paragraph.

Applicant has amended claims 1, 22 and 43 to emphasize the use of binary values of bits in the context of assigning values to elements (bits) of a bit array.

Applicant regards the "synchronization of bit position" as accurately describing the invention. As described in further detail below, the bit position is related to a "rank" of a record within a lookup table. Therefore, Applicant maintains the claim language "synchronizing the bit position" and respectfully submits such language is proper.

With regard to dependent claims 2-15, 23-36 and 44-57, Applicant respectfully submits the claims 2-15, 23-36 and 44-57 use the phrase "binary digits" to properly refer to bits when applicable.

With regard to claims 19, 40 and 61, Applicant respectfully submits that in the context of a logical bit-wise "AND" operation, as described in prior art, the "AND" operation would return a result if either of a first or second record values is encountered. However, in the context of the invention, namely because the invention utilizes a "rank" value associated with a record to index a bit position in the bit array, the "AND" operation will exclusively return those records that have both the first and the second values.

With regard to claims 8, 24 and 50, Applicant has amended claims 2, 8, 23, 44 and 50 to both clarify the claim language and the dependency references.

With regard to claims 14, 35 and 56, Applicant has corrected the dependency references, which results in correcting the antecedent basis issues.

With regard to claim 48, Applicant has amended the claim to correct the dependency reference, which results in correcting the antecedent basis issue.

IV. Rejection of Claims 16-18, 37-39 and 58-60 Based on 35 U.S.C. § 102

The Examiner has rejected claims 16-18, 37-39 and 58-60 under 35 U.S.C. § 102(b) as being anticipated by O'Neil *et al.* ("Improved Query Performance with Variant Indexes").

Applicant respectfully disagrees and submits that O'Neil does not anticipate claims 16-18, 37-39 and 58-60 as amended, and that claims 16-18, 37-39 and 58-60 are allowable for at least the reasons detailed below.

A. Rejection of Independent Claims 16, 37 and 58

Applicant respectfully submits that independent claims 16, 37 and 58, as amended, are allowable because O'Neil does not anticipate, teach or suggest creating a first bit vector representation for a first combination of field values utilizing at least two lookup tables, creating a second bit vector representation for a second combination of field values utilized as a search bit vector and performing a bit-level operation on the first and second bit vector representations for which a result other than "0" indicates one or more data records found.

O'Neil's paper presents a synthesis of existing techniques for performing bit-wise indexing and searching through a relational database using bit vector representations of the

data fields. None of the cited techniques describe, or teach utilizing a unique numerical value associated with each value of a field in a database table, and using the numerical value's value (or content) as a pointer in a bit array. Nor does O'Neil utilize lookup tables to order (e.g. sort or rank) values from a specific field in a database table, associate each of the values of the field with a unique numerical value that is used to point to a position in the array of bits (the bit vector). In fact, an application using any of the prior art techniques (implementing bitmap indexing) described in O'Neil, would provide a search using bit-level operations on single field level and not on combinations of values from different fields in a single bit-wise operation.

Therefore, O'Neil does not anticipate, teach or suggest creating a first bit vector representation for a first combination of field values utilizing at least two lookup tables, creating a second bit vector representation for a second combination of field values utilized as a search bit vector and performing a bit-level operation

B. Rejection of Dependent Claims 17, 18, 38, 39, 59 and 60

Applicant submits that claims 17, 18, 38, 39, 59 and 60 being dependent upon respective allowable base claims are also allowable for at least the foregoing reasons stated above.

V. Rejection of Claims 1 through 63 Based on 35 U.S.C. § 103 (a)

The Examiner has rejected claims 1-15, 19-36, 40-57 and 61-63 under 35 U.S.C. § 103(a) as being unpatentable over O'Neil *et al.* ("Improved Query Performance with Variant Indexes") in view of Deplege et al. (U.S. Patent 5,884,307), and claims 16-18, 37-

39 and 58-60 over O'Neil *et al.*, as applied to claims above, and further in view of Schildt ("C: The Complete Reference").

Applicant respectfully disagrees and submits that claims 1-15, 19-36, 40-57 and 61-63, as well as claims 16-18, 37-39 and 58-60, as amended, are allowable for at least the reasons detailed below.

A. Rejection of Independent Claims 1, 16, 22, 37, 43, 58

Applicant respectfully disagrees and submits that independent claims 1, 22 and 43, as amended, are allowable because O'Neil, alone or combined with Depledge *et al.* does not teach, or suggest claims 1, 22 and 43; neither does O'Neil, alone or combined with Schildt teach, suggest or describe independent claims 16, 37 and 58.

With regard to independent claims 1, 22 and 43, O'Neil, alone or combined with Depledge *et al.* does not teach, suggest or describe building sets of values for association with numerical values so that each numerical value can be used to point to a specific position in a bit array, allowing for encoding of entire data records in an array, and for searching for specific combinations of values in bit vectors using a simple bit-level "AND" operation. The invention associates a bit vector representation with each value encountered in data records of a table using a lookup table. Each lookup table contains an association of the value with an identification value indicating a bit position in the bit vector representation. The numerical values are then combined to form an array of bits that encode each data record.

O'Neil's paper presents a synthetic overview of bit mapping techniques used for indexing data records in a relational database. Bitmapping relies on (see page 39, 2nd paragraph of column 2) "A Bitmap B is defined on T as a sequence of M bits. If a Bitmap B is meant to list rows in T with a given property P, then for each row r with row number j that has the property P, we set bit j in B to one;". With the latter approach, the bit locations indicate the number of those records that have a specific value for a given field. In O'Neil

indexing a table field relies on using strings of bits where each bit reflects whether a row that has a given property.

The present invention associates each value of a field with a specific numerical value. The numerical value is then utilized as a position indicator in a segment allocated to the property in a bit string. The result is a bit string representing each data record. An application implementing the invention first consults the lookup table to build search bit string. The search bit string is then "AND"ed with each bit string representation to determine the data records that match the search bit string.

Therefore, O'Neil, alone or combined with either Depledge *et al.* or Schildt does not teach, or suggest independent claims 1, 16, 22, 37, 43 and 58.

B. Rejection of Dependent Claims 2-15, 17-21, 23-36, 38-42, 44-57 and 59-63

Applicant submits that claims 2-15, 17-21, 23-36, 38-42, 44-57 and 59-63 being dependent upon respective allowable base claims are also allowable for at least the foregoing reasons stated above.

CONCLUSION

For at least the foregoing reasons, Applicant respectfully submits that pending claims 1-63, as amended, are patentably distinct from the prior art of record and in condition for allowance. Applicant therefore respectfully requests that pending claims 1-63 be allowed.

Respectfully submitted,

THE HECKER LAW GROUP

Date: 7/17/03

By: Obi I. Iloputaife
Reg. No. 45,677

THE HECKER LAW GROUP
1925 Century Park East
Suite 2300
Los Angeles, California 90067
(310) 286-0377

CERTIFICATE OF MAILING

This is to certify that this correspondence is being deposited with the United States Postal Service with sufficient postage as First Class Mail in an envelope addressed to Commissioner for Patents, P.O. Box 1450 Alexandria, VA 22313-1450 on July 17, 2003

MKiran 7-17-03

Siganture: Mona Kiran *Date*

AMENDMENTS TO THE CLAIMS

What is claimed is:

SUB B27

1. (NEWLY AMENDED) A method of indexing occurrences of a value in at least one data record using a bit vector representation, the method comprising:

associating a bit vector representation with a plurality of values encountered in at least two data fields;

associating a bit position of the bit vector representation to the at least one record in a lookup table that contains an association of the value with an identification value utilized to point to the bit position;

determining whether one of the plurality of values exists in the at least one data record; and

assigning a binary value to the bit position in the bit vector representation based on the outcome of the determining step;

synchronizing the bit position with the one of the plurality of values to reflect any updates to the value.

A1

2. (NEWLY AMENDED) A method according to Claim 1, further comprising:

encoding the bit vector representation to produce an encoded bit representation.

3. A method according to Claim 2, wherein the bit vector representation comprises a sequence of bits, encoding the bit vector representation further comprising:

determining whether a frequency of a binary digit is less than a threshold value; and

storing as the encoded bit vector representation at least one position of the binary digit in the bit vector representation.

4. A method according to Claim 3, wherein the threshold is the number of bits used to store a number.

5. A method according to Claim 2, wherein the bit vector representation comprises a sequence of binary digits, encoding the bit vector representation further comprising:

determining whether a size of a region of like binary digits is greater than a threshold value; and storing as the encoded bit vector representation a representation of the region.

6. A method according to claim 5, wherein the representation comprises a start and end designation pair that represents the start and ending bits of the region.

7. A method according to Claim 5, wherein the threshold is twice the number of bits used to store a number.

8. (NEWLY AMENDED) A method according to Claim 2, further comprising:

compressing the encoded bit vector representation.

9. A method according to Claim 1, wherein the bit vector representation is compressed using a compression technique.

A1

10. A method according to Claim 1, wherein the bit vector representation is encoded and compressed.

11. A method according to Claim 1, wherein the data structure is a record in a database.

12. A method according to Claim 1, further comprising:

examining the bit vector representation to determine whether the data record contains the value.

13. A method according to Claim 1, wherein plural bit vector representations exist each representing a discrete value, the method further comprising:

determining whether the data record contains more than one of the values by performing a bit-level operation on the corresponding bit vector representations.

Version with marking to show changes

14. (NEWLY AMENDED) A method according to Claim 13, wherein the bit-level operation is an "OR" operation.

15. A method according to Claim 13, wherein the operation is an "AND" operation.

16. (NEWLY AMENDED) A method of identifying combinations of values used in at least one data record comprising fields for storing the values, the method comprising:

creating a first bit vector representation for a first combination of field values, the first bit vector representation identifying use of the first combination of field values in at least one data record utilizing at least two lookup tables;

creating a second bit vector representation for a second combination of field values, the second bit vector representation identifying use of a search for the first combination of field values in at least one data record; and

performing a bit-level operation on the first and second bit vector representations.

A1

17. A method according to Claim 16, wherein the bit-level operation is an "AND" operation.

18. A method according to Claim 17, wherein the "AND" operation is a bit-wise "AND" returning a bit corresponding to each of the at least one data record identifying whether a combination of the first and second values exist in at least one data record.

19. A method according to Claim 17, wherein the "AND" operation is a logical "AND" returning a single result representing whether any of the at least one data record contains a combination of the first and second values.

20. A method according to Claim 16, further comprising:

updating the at least one data record.

21. A method according to Claim 20, wherein the update to the at least one data record changes the first value, the method further comprising:

updating the first bit vector representation to reflect the update to the at least one data record.

22. (NEWLY AMENDED) A computer-readable memory medium in which computer-executable process steps are stored, the process steps for indexing occurrences of a value in at least one data record using a bit vector representation, wherein the process steps comprise:

an associating step to associate a bit vector representation with a plurality of values encountered in at least two data fields;

an associating step to associate a bit position of the bit vector representation to the at least one record in a lookup table that contains an association of the value with an identification value utilized to point to the bit position;

a determining step to determine whether one of the plurality of values exists in the at east one data record; and

an assigning step to assign a binary value to the bit position in the bit vector representation based on the outcome of the determining step;

a synchronizing step to synchronize the bit position with one of the plurality of values to reflect any updates to the value.

A1

23. (NEWLY AMENDED) A computer-readable memory medium according to Claim 22, further comprising:

an encoding step to encode the bit vector representation to produce an encoded bit representation.

24. A computer-readable memory medium according to Claim 23, wherein the bit vector representation comprises a sequence of bits, encoding the bit vector representation further comprising:

a determining step to determine whether a frequency of a binary digit is less than a threshold value; and

a storing step to store as the encoded bit vector representation at least one position of the binary digit in the bit vector representation.

25. A computer-readable memory medium according to Claim 24, wherein the threshold is the number of bits used to store a number.

26. A computer-readable memory medium according to Claim 23, wherein the bit vector representation comprises a sequence of binary digits, encoding the bit vector representation further comprising:

a determining step to determine whether a size of a region of like binary digits is greater than a threshold value; and

a storing step to store as the encoded bit vector representation a representation of the region.

27. A computer-readable memory medium according to claim 26, wherein the representation comprises a start and end designation pair that represents the start and ending bits of the region.

28. A computer-readable memory medium according to Claim 26, wherein the threshold is twice the number of bits used to store a number.

29. A computer-readable memory medium according to Claim 22, further comprising:

a compressing step to compress the encoded bit vector representation.

30. A computer-readable memory medium

according to Claim 22, wherein the bit vector representation is compressed using a compression technique.

31. A computer-readable memory medium

according to Claim 22, wherein the bit vector representation is encoded and compressed.

32. A computer-readable memory medium according to Claim 22, wherein the data structure is a record in a database.

33. A computer-readable memory medium according to Claim 22, further comprising:

an examining step to examine the bit vector representation to determine whether the data record contains the value.

34. A computer-readable memory medium according to Claim 22, wherein plural bit vector representations exist each representing a discrete value, the method further comprising:

a determining step to determine whether the data record contains more than one of the values by performing a bit-level operation on the corresponding bit vector representations.

35. (NEWLY AMENDED) A computer-readable memory medium according to Claim 34, wherein the bit-level operation is an "OR" operation.

36. A computer-readable memory medium according to Claim 34, wherein the operation is an "AND" operation.

37. (NEWLY AMENDED) A computer-readable memory medium in which computer-executable process steps are stored, the process steps for identifying combinations of values used in at least one data record comprising fields for storing the values, wherein the process steps comprise:

a creating step to create a first bit vector representation for a first combination of field values, the first bit vector representation identifying use of the first combination of field values in the at least one data record utilizing at least two lookup tables;

a creating step to create a second bit vector representation for a second combination of field values, the second bit vector representation identifying use of a search for the first combination of field values in the at least one data record; and

a performing step to perform a bit-level operation on the first and second bit vector representations.

38. A computer-readable memory medium according to Claim 37, wherein the bit-level operation is an "AND" operation.

39. A computer-readable memory medium according to Claim 38, wherein the "AND" operation is a logical "AND" returning a bit corresponding to each of the at least one data record identifying whether a combination of the first and second values exist in at least one data record.

40. A computer-readable memory medium according to Claim 38, wherein the "AND" operation is a bit-wise "AND" returning a single result representing whether any of the at least one data record contains a combination of the first and second values.

41. A computer-readable memory medium according to Claim 37, further comprising:

an updating step to update the at least one data record.

42. (NEWLY AMENDED) A computer-readable memory medium according to Claim 41, further comprising:

a determining step to determine whether the update to the at least one data record affects the first bit vector representation;

an updating step to update the first bit vector representation, if it is determined that the update to the at least one data record affects the first bit vector representation;

a determining step to determine whether the update to the at least one data record affects the second bit vector representation; and

an updating step to update the second bit vector representation, if it is determined that the update to the at least one data record affects the second bit vector representation.

43. (NEWLY AMENDED) Computer-executable process steps

stored on a computer readable medium, said computer-executable process steps for indexing occurrences of a value in at least one data record using a bit vector representation, said computer-executable process steps comprising:

code to associate a bit vector representation with plurality of values encountered in at least two data fields;

code to associate a bit position of the bit vector representation to the at least one record in a lookup table that contains an association of the the value with an identification value utilized to point to the bit position;

code to determine whether one of the plurality of values exists in at least one data record;

code to assign a binary value to the bit position in the bit vector representation based on the outcome of the determining step;

code to synchronize the bit position with the one of the plurality of values to reflect any updates to the value.

44. (NEWLY AMENDED) A computer-executable process steps according to Claim 43, further comprising:

code to encode the bit vector representation producing an encoded bit representation.

45. A computer-executable process steps according to Claim 44, wherein the bit vector representation comprises a sequence of bits, encoding the bit vector representation further comprising:

code to determine whether a frequency of a binary digit is less than a threshold value; and

code to store as the encoded bit vector representation at least one position of the binary digit in the bit vector representation.

46. A computer-executable process steps according to Claim 45, wherein the threshold is the number of bits used to store a number.

47. A computer-executable process steps according to Claim 44, wherein the bit vector representation comprises a sequence of binary digits, encoding the bit vector representation further comprising:

code to determine whether a size of a region of like binary digits is greater than a threshold value; and

code to store as the encoded bit vector representation a representation of the region.

48. (NEWLY AMENDED) A computer-executable process steps according to claim 47, wherein the representation comprises a start and end designation pair that represents the start and ending bits of the region.

49. A computer-executable process steps according to Claim 47, wherein the threshold is twice the number of bits used to store a number.

50. (NEWLY AMENDED) A computer-executable process steps according to Claim 44, further comprising:

code to compress the encoded bit vector representation.

51. A computer-executable process steps according to Claim 43, wherein the bit vector representation is compressed using a compression technique.

52. A computer-executable process steps according to Claim 43, wherein the bit vector representation is encoded and compressed.

53. A computer-executable process steps according to Claim 43, wherein the data structure is a record in a database.

54. A computer-executable process steps according to Claim 43, further comprising:

code to examine the bit vector representation to determine whether the data record contains the value.

A1

55. A computer-executable process steps according to Claim 43, wherein plural bit vector representations exist each representing a discrete value, the method further comprising:

code to determine whether the data record contains more than one of the values by performing a bit-level operation on the corresponding bit vector representations.

56. (NEWLY AMENDED) A computer-executable process steps according to Claim 51, wherein the bit-level operation is an "OR" operation.

57. A computer-executable process steps

according to Claim 55, wherein the operation is an "AND" operation.

58. (NEWLY AMENDED) Computer-executable process steps stored on a computer readable medium, said computer-executable process steps for identifying combinations of values used in at least one data record comprising fields for storing the values, said computer-executable process steps comprising:

code to create a first bit vector representation for a first combination of field values, the first bit vector representation identifying use of the first combination of field values in at least one data record utilizing at least two lookup tables;

code to create a second bit vector representation for a second combination of field values, the second bit vector representation identifying use of a search for the first combination of field values in at least one data record; and

code to perform a bit-level operation on the first and second bit vector representations.

59. A computer-executable process steps according to Claim 58, wherein the bit-level operation is an "AND" operation.

60. A computer-executable process steps according to Claim 59, wherein the "AND" operation is a logical "AND" returning a bit corresponding to each of the at least one data record identifying whether a combination of the first and second values exist in at least one data record.

A computer-executable process steps according to Claim 59, wherein the "AND" operation is a bit-wise "AND" returning a single result representing whether any of the at least one data record contains a combination of the first and second values.

62. A computer-executable process steps according to Claim 58, further comprising:

code to update the at least one data record.

63. (NEWLY AMENDED) A computer-executable process steps according to Claim 62, further comprising:

code to determine whether the update to the at least one data record should effect the first bit vector representation;

code to update the first bit vector representation, if it is determined that the update to the at least one data record should effect the first bit vector representation;

code to determine whether the update to the at least one data record should effect the second bit vector representation; and

code to update the second bit vector representation, if it is determined that the update to the at least one data record should effect the second bit vector representation.

Version with marking to show changes

Claims

What is claimed is:

1. (NEWLY AMENDED) A method of indexing occurrences of a value in at least one data record using a bit vector representation, the method comprising:

associating a bit vector representation with a ~~value~~plurality of values encountered in at least two data fields;

associating a bit position of the bit vector representation to the at least one record in a lookup table that contains an association of the the value with an indentification value utilized to point to the bit position;

determining whether one of the plurality of values ~~the value~~ exists in the at least one data record; and

assigning a binary value to the bit position in the bit vector representation based on the outcome of the determining step;

synchronizing the bit position with the one of the plurality of values ~~the value~~ to reflect any updates to the value.

2. (NEWLY AMENDED) A method according to Claim 1, further comprising:

encoding the bit vector representation to produce an encoded bit representation.

3. A method according to Claim 2, wherein the bit vector representation comprises a sequence of bits, encoding the bit vector representation further comprising:

determining whether a frequency of a binary digit is less than a threshold value; and

storing as the encoded bit vector representation at least one position of the binary digit in the bit vector representation.

4. A method according to Claim 3, wherein the threshold is the number of bits used to store a number.

5. A method according to Claim 2, wherein the bit vector representation comprises a sequence of binary digits, encoding the bit vector representation further comprising:

determining whether a size of a region of like binary digits is greater than a threshold value; and

storing as the encoded bit vector representation a representation of the region.

6. A method according to claim 5, wherein the representation comprises a start and end designation pair that represents the start and ending bits of the region.

7. A method according to Claim 5, wherein the threshold is twice the number of bits used to store a number.

8. (NEWLY AMENDED) A method according to Claim ~~1~~2, further comprising:

compressing the encoded bit vector representation.

9. A method according to Claim 1, wherein the bit vector representation is compressed using a compression technique.

10. A method according to Claim 1, wherein the bit vector representation is encoded and compressed.

11. A method according to Claim 1, wherein the data structure is a record in a database.

12. A method according to Claim 1, further comprising:

examining the bit vector representation to determine whether the data record contains the value.

13. A method according to Claim 1, wherein plural bit vector representations exist each representing a discrete value, the method further comprising:

determining whether the data record contains more than one of the values by performing a bit-level operation on the corresponding bit vector representations.

14. (NEWLY AMENDED) A method according to Claim ~~9~~13, wherein the bit-level operation is an "OR" operation.

15. A method according to Claim 13, wherein the operation is an "AND" operation.

16. (NEWLY AMENDED) A method of identifying combinations of values used in at least one data record comprising fields for storing the values, the method comprising:

creating a first bit vector representation for a first combination of field values~~value~~, the first bit vector representation identifying use of the first combination of field values~~value~~ in at least one data record utilizing at least two lookup tables;

creating a second bit vector representation for a second combination of field values~~value~~, the second bit vector representation identifying use of ~~the~~ a search for the first combination of field values~~second value~~ in at least one data record; and

performing a bit-level operation on the first and second bit vector representations.

17. A method according to Claim 16, wherein the bit-level operation is an "AND" operation.

18. A method according to Claim 17, wherein the "AND" operation is a bit-wise "AND" returning a bit corresponding to each of the at least one data record identifying whether a combination of the first and second values exist in at least one data record.

19. A method according to Claim 17, wherein the "AND" operation is a logical "AND" returning a single result representing whether any of the at least one data record contains a combination of the first and second values.

20. A method according to Claim 16, further comprising:

updating the at least one data record.

21. A method according to Claim 20, wherein the update to the at least one data record changes the first value, the method further comprising:

updating the first bit vector representation to reflect the update to the at least one data record.

22. (NEWLY AMENDED) A computer-readable memory medium in which computer-executable process steps are stored, the process steps for indexing occurrences of a value in at least one data record using a bit vector representation, wherein the process steps comprise:

an associating step to associate a bit vector representation with a ~~value~~plurality of values encountered in at least two data fields;

an associating step to associate a bit position of the bit vector representation to the at least one record in a lookup table that contains an association of the the value with an indentification value utilized to point to the bit position;

a determining step to determine whether one of the plurality of values ~~the value~~ exists in the at east one data record; and

an assigning step to assign a binary value to the bit position in the bit vector representation based on the outcome of the determining step;

a synchronizing step to synchronize the bit position with one of the plurality of values ~~the value~~ to reflect any updates to the value.

23. (NEWLY AMENDED) A computer-readable memory medium according to Claim 22, further comprising:

an encoding step to encode the bit vector representation to produce an encoded bit representation.

24. A computer-readable memory medium according to Claim 23, wherein the bit vector representation comprises a sequence of bits,encoding the bit vector representation further comprising:

a determining step to determine whether a frequency of a binary digit is less than a threshold value; and

a storing step to store as the encoded bit vector representation at least one position of the binary digit in the bit vector representation.

25. A computer-readable memory medium according to Claim 24, wherein the threshold is the number of bits used to store a number.

26. A computer-readable memory medium according to Claim 23, wherein the bit vector representation comprises a sequence of binary digits, encoding the bit vector representation further comprising:

a determining step to determine whether a size of a region of like binary digits is greater than a threshold value; and

a storing step to store as the encoded bit vector representation a representation of the region.

27. A computer-readable memory medium according to claim 26, wherein the representation comprises a start and end designation pair that represents the start and ending bits of the region.

28. A computer-readable memory medium according to Claim 26, wherein the threshold is twice the number of bits used to store a number.

29. A computer-readable memory medium according to Claim 22, further comprising:

a compressing step to compress the encoded bit vector representation.

30. A computer-readable memory medium

according to Claim 22, wherein the bit vector representation is compressed using a compression technique.

31. A computer-readable memory medium

according to Claim 22, wherein the bit vector representation is encoded and compressed.

32. A computer-readable memory medium according to Claim 22, wherein the data structure is a record in a database.

33. A computer-readable memory medium according to Claim 22, further comprising:

an examining step to examine the bit vector representation to determine whether the data record contains the value.

34. A computer-readable memory medium according to Claim 22, wherein plural bit vector representations exist each representing a discrete value, the method further comprising:

a determining step to determine whether the data record contains more than one of the values by performing a bit-level operation on the corresponding bit vector representations.

35. (NEWLY AMENDED) A computer-readable memory medium according to Claim ~~30~~34, wherein the bit-level operation is an "OR" operation.

36. A computer-readable memory medium according to Claim 34, wherein the operation is an "AND" operation.

37. (NEWLY AMENDED) A computer-readable memory medium in which computer-executable process steps are stored, the process steps for identifying combinations of values used in at least one data record comprising fields for storing the values, wherein the process steps comprise:

a creating step to create a first bit vector representation for a first combination of field values~~value~~, the first bit vector representation identifying use of the first combination of field values~~value~~ in the at least one data record utilizing at least two lookup tables;

a creating step to create a second bit vector representation for a second combination of field values~~value~~, the second bit vector representation identifying use of ~~the~~ a search for the first combination of field values~~econd value~~ in the at least one data record; and

a performing step to perform a bit-level operation on the first and second bit vector representations.

38. A computer-readable memory medium according to Claim 37, wherein the bit-level operation is an "AND" operation.

39. A computer-readable memory medium according to Claim 38, wherein the "AND" operation is a logical "AND" returning a bit corresponding to each of the at least one data record identifying whether a combination of the first and second values exist in at least one data record.

40. A computer-readable memory medium according to Claim 38, wherein the "AND" operation is a bit-wise "AND" returning a single result representing whether any of the at least one data record contains a combination of the first and second values.

41. A computer-readable memory medium according to Claim 37, further comprising:

an updating step to update the at least one data record.

42. (NEWLY AMENDED) A computer-readable memory medium according to Claim 41, further comprising:

a determining step to determine whether the update to the at least one data record ~~effects~~affects the first bit vector representation;

an updating step to update the first bit vector representation, if it is determined that the update to the at least one data record ~~effects~~affects the first bit vector representation;

a determining step to determine whether the update to the at least one data record ~~effects~~affects the second bit vector representation; and

an updating step to update the second bit vector representation, if it is determined that the update to the at least one data record ~~effects~~affects the second bit vector representation.

43. (NEWLY AMENDED) Computer-executable process steps

stored on a computer readable medium, said computer-executable process steps for indexing occurrences of a value in at least one data record using a bit vector representation, said computer-executable process steps comprising:

code to associate a bit vector representation with ~~value~~plurality of values encountered in at least two data fields;

code to associate a bit position of the bit vector representation to the at least one record in a lookup table that contains an association of the the value with an indentification value utilized to point to the bit position;

code to determine whether one of the plurality of values ~~the value~~ exists in at least one data record;

code to assign a binary value to the bit position in the bit vector representation based on the outcome of the determining step;

code to synchronize the bit position with the one of the plurality of values ~~the value~~ to reflect any updates to the value.

44. (NEWLY AMENDED) A computer-executable process steps according to Claim 43, further comprising:

code to encode the bit vector representation producing an encoded bit representation.

45. A computer-executable process steps according to Claim 44, wherein the bit vector representation comprises a sequence of bits, encoding the bit vector representation further comprising:

code to determine whether a frequency of a binary digit is less than a threshold value; and

code to store as the encoded bit vector representation at least one position of the binary digit in the bit vector representation.

46. A computer-executable process steps according to Claim 45, wherein the threshold is the number of bits used to store a number.

47. A computer-executable process steps according to Claim 44, wherein the bit vector representation comprises a sequence of binary digits, encoding the bit vector representation further comprising:

code to determine whether a size of a region of like binary digits is greater than a threshold value; and

code to store as the encoded bit vector representation a representation of the region.

48. (NEWLY AMENDED) A computer-executable process steps according to claim ~~50~~47, wherein the representation comprises a start and end designation pair that represents the start and ending bits of the region.

49. A computer-executable process steps according to Claim 47, wherein the threshold is twice the number of bits used to store a number.

50. (NEWLY AMENDED) A computer-executable process steps according to Claim ~~43~~44, further comprising:

code to compress the encoded bit vector representation.

51. A computer-executable process steps according to Claim 43, wherein the bit vector representation is compressed using a compression technique.

52. A computer-executable process steps according to Claim 43, wherein the bit vector representation is encoded and compressed.

53. A computer-executable process steps according to Claim 43, wherein the data structure is a record in a database.

54. A computer-executable process steps according to Claim 43, further comprising:

code to examine the bit vector representation to determine whether the data record contains the value.

55. A computer-executable process steps according to Claim 43, wherein plural bit vector representations exist each representing a discrete value, the method further comprising:

code to determine whether the data record contains more than one of the values by performing a bit-level operation on the corresponding bit vector representations.

56. (NEWLY AMENDED) A computer-executable process steps according to Claim ~~55~~51, wherein the bit-level operation is an "OR" operation.

57. A computer-executable process steps
according to Claim 55, wherein the operation is an "AND" operation.

58. (NEWLY AMENDED) Computer-executable process steps stored on a computer readable medium, said computer-executable process steps for identifying combinations of values used in at least one data record comprising fields for storing the values, said computer-executable process steps comprising:

code to create a first bit vector representation for a first combination of field values~~value~~, the first bit vector representation identifying use of the first combination of field values~~value~~ in at least one data record utilizing at least two lookup tables;

code to create a second bit vector representation for a second combination of field values~~value~~, the second bit vector representation identifying use of ~~the~~ a search for the first combination of field values~~second value~~ in at least one data record; and

code to perform a bit-level operation on the first and second bit vector representations.

59. A computer-executable process steps according to Claim 58, wherein the bit-level operation is an "AND" operation.

60. A computer-executable process steps according to Claim 59, wherein the "AND" operation is a logical "AND" returning a bit corresponding to each of the at least one data record identifying whether a combination of the first and second values exist in at least one data record.

A computer-executable process steps according to Claim 59, wherein the "AND" operation is a bit-wise "AND" returning a single result representing whether any of the at least one data record contains a combination of the first and second values.

62. A computer-executable process steps according to Claim 58, further comprising:

code to update the at least one data record.

63. (NEWLY AMENDED) A computer-executable process steps according to Claim 62, further comprising:

code to determine whether the update to the at least one data record ~~effects~~should effect the first bit vector representation;

code to update the first bit vector representation, if it is determined that the update to the at least one data record ~~effects~~should effect the first bit vector representation;

code to determine whether the update to the at least one data record ~~effects~~should effect the second bit vector representation; and

code to update the second bit vector representation, if it is determined that the update to the at least one data record ~~effects~~should effect the second bit vector representation.